شرح برنامجٍ لحل المعادلة التربيعية باستخدام لغة "إبداع"

م. وائل حسن محمد – أبو إياس

في هذا الكُتيِّب سأقوم بشرحٍ برنامجٍ بسيطٍ لحل المعادلة التربيعية و إيجاد جذريها سواءٌ أكانا حقيقيين أم جذرين تخيليين. و بالطبع سيكون البرنامج مكتوباً بِلُغةإبداع. و سأستغل هذا الشرح لتقديم فكرةٍ بسيطةٍ عن بعض قواعد إبداع و كذلك لتوضيح مدي سهولة البرمجة بها و سهولة تحويل الخوارزم algorithm إلي برنامجٍ مكتوبٍ باستخدامها.

و لا يتطلب هذا الشرح إلا القدر الأدني من المعرفة البرمجية، لكن المبتدئ الصرف الذي لم يتعلم أي شيءٍ عن أية لغة برمجةٍ من قبل ربما لا يستوعب إلا نذراً قليلاً منه.

***

المطلوب هو إيجاد الجذرين الخاصين بالمعادلة التربيعية التي تكون علي الشكل التالي:

أ س 2 + ب س + ج = 0

حيث (أ) و (ب) و (ج) هن مُعامِلاتٌ لحدود المعادلة، و القانونان الخاصان بإيجاد الجذرين سهلان جداً و مباشران للغاية، حيث:

الجذر الأول = (- ب + **الجذر التربيعي للقيمة** (ب 2 − 4 × أ × ج)) ÷ (2 × أ)

الجذر الثاني = (- ب - **الجذر التربيعي للقيمة** (ب 2 − 4 × أ × ج)) ÷ (2 × أ)

مثال:

المعادلة (2 س 2 + 3 س + 1 = 0) لها الجذران:

الجذر الأول = -0.5

الجذر الثاني = -1

الآن تعالوا نكتب خطوات الحل بالترتيب (خوارزم الحل):

في البداية علينا أن ننتبه إلي أنه من الممكن كتابة البرنامج الذي يحل هذه المشكلة في أسطرٍ قليلةٍ جداً؛ حيث أنه يمكن جعل

خطوات الحل كما يلي:

- **قراءة المُعامِل أ.**
- **قراءة المُعامِل ب.**
- **قراءة المُعامِل ج.**
- **حساب الجذر الأول عن طريق التعويض مباشرةً في القانون الخاص به.**
- **حساب الجذر الثاني عن طريق التعويض مباشرةً في القانون الخاص به.**
- **كتابة قيمتي الجذرين علي الشاشة.**

لكن المشكلة أن الجذور لن تكون دائماً علي شكل أعدادٍ بسيطة؛ فمن الممكن أن تكون الجذور تخيلية، خذ المعادلة التربيعية التالية كمثالٍ لهذا الأمر:

س 2 + س + 9 = 0

فقيمتا الجذرين هما:

الجذر الأول = -0.5 + 2.958039891549808 ت

الجذر الثاني = -0.5 − 2.958039891549808 ت

و هذا يرجع إلي أن القيمة (ب 2 − 4 × أ × ج) قيمةٌ سالبةٌ لأن قيمة ب 2 أقل من قيمة 4 × أ × ج.

و لأن الحاسوب لا يعرف إلا ما نخبره به فإنه سيقوم بالتعويض في المعادلتين السابقتين و حينما يُحاوِل إيجاد قيمة الجذر لعددٍ سالبٍ سيحدث خطأ زمن تشغيل runtime error لأنه ليس هناك جذرٌ رقميٌ بسيطٌ للأعداد السالبة، و إنما يوجد لها جذورٌ تخيلية (لو لم تكن تعلم ما هي الجذور التخيلية فأنت بحاجة لمراجعةٍ بسيطةٍ عنها).

إذاً ما العمل الآن ؟**!**

بسيطة؛ يمكننا تحويل الخوارزم إلي الشكل التالي:

•قراءة المُعامِل أ.

•قراءة المُعامِل ب.

•قراءة المُعامِل ج.

•حساب القيمة التي تحت الجذر (ب 2 − 4 × أ × ج):

•لو كانت القيمة موجبةً فالجذران حقيقيان و يمكننا التكملة كما سبق:

•حساب الجذر الأول عن طريق التعويض مباشرةً في قانونه.

•حساب الجذر الثاني عن طريق التعويض مباشرةً في قانونه.

•أما لو كانت القيمة سالبةً فالجذران تخيليان، و يتم حسابهما عن طريق المعادلتين:

الجذر الأول = (-ب ÷ 2 × أ) + (جذر القيمة (- القيمة التي تحت الجذر)) ت

الجذر الثاني = (-ب ÷ 2 × أ) - (جذر القيمة (- القيمة التي تحت الجذر)) ت

•كتابة قيمتي الجذرين علي الشاشة بشكلٍ مناسبٍ لنوع الجذور.

هذا جميل، و لكن ماذا لو أدخل المستخدم (سواءٌ عمداً أم عن طريق الخطأ) قيمة المُعامِل (أ) بصفر ؟**!**

هذا موقفٌ يجب التحسب له أيضاً، و لكن كيف يكون هذا ؟

سنلجأ لاختبار قيمة المُعامِل (أ) قبل البدء بالحل، و لو كان صفراً فسوف نعتبر أن المعادلة من الدرجة الأولي و ليست تربيعية، و بالتالي لها جذرٌ واحدٌ فقط يتم حساب قيمته من خلال القانون:

الجذر = -ج ÷ ب

و هكذا يتحول الخوارِزم إلي الشكل التالي:

•قراءة المُعامِل أ.

•قراءة المُعامِل ب.

•**قراءة المُعامِل ج.**

•**اختبار كون المُعامِل أ يساوي صفر:**

•**لو كان أ = صفر:**

•**اعتبر المعادلة من الدرجة الأولي و احسب قيمة الجذر بالمعادلة (الجذر = -ج ÷ ب).**

•**اكتب الرقم علي الشاشة.**

•**لو لم يكن أ يساوي 0 :**

•**حساب القيمة التي تحت الجذر (ب 2 − 4 × أ × ج):**

•**لو كانت القيمة موجبةً فالجذران حقيقيان و يمكننا التكملة كما سبق:**

•**حساب الجذر الأول عن طريق التعويض مباشرةً في قانونه.**

•**حساب الجذر الثاني عن طريق التعويض مباشرةً في قانونه.**

•**أما لو كانت القيمة سالبةً فالجذران تخيليان، و يتم حسابهما عن طريق المعادلتين:**

الجذر الأول = (-ب ÷ 2 × أ) + (جذر القيمة (- القيمة التي تحت الجذر)) ت

الجذر الثاني = (-ب ÷ 2 × أ) - (جذر القيمة (- القيمة التي تحت الجذر)) ت

•**كتابة قيمتي الجذرين علي الشاشة بشكلٍ مناسبٍ لنوع الجذور.**

لكن ألا تلاحظون أن هذا الخوارزم لا يحل إلا معادلةً واحدةً ثم يتوقف التنفيذ بعدها ؟

إذاً فهو يحتاج إلي إضافةٍ اخري ليُمكِن من خلاله حل أي عددٍ نرغب في حله من المعادلات التربيعية، و سيصبح شكله بعدها كما يلي:

•**بينما هناك معادلة نرغب في حلها:**

•**قراءة المُعامِل أ.**

•**قراءة المُعامِل ب.**

•**قراءة المُعامِل ج.**

•**اختبار كون المُعامِل أ يساوي صفر:**

•**لو كان أ = صفر :**

•**اعتبر المعادلة من الدرجة الأولي و احسب قيمة الجذر بالمعادلة (الجذر= -ج ÷ ب).**

•**اكتب الرقم علي الشاشة.**

•**لو لم يكن أ يساوي 0 :**

•**حساب القيمة التي تحت الجذر (ب 2 – 4 × أ × ج):**

•**لو كانت القيمة موجبةً فالجذران حقيقيان و يمكننا التكملة كما سبق:**

•**حساب الجذر الأول عن طريق التعويض مباشرةً في قانونه.**

•**حساب الجذر الثاني عن طريق التعويض مباشرةً في قانونه.**

•**أما لو كانت القيمة سالبةً فالجذران تخيليان، و يتم حسابهما عن طريق المعادلتين:**

الجذر الأول = (-ب ÷ 2 × أ) + (جذر القيمة (-القيمة التي تحت الجذر)) ت

الجذر الثاني = (-ب ÷ 2 × أ) - (جذر القيمة (-القيمة التي تحت الجذر)) ت

•**كتابة قيمتي الجذرين علي الشاشة بشكلٍ مناسبٍ لنوع الجذور.**

و هكذا نكون قد حللنا كل مشاكل الخوارزم و أصبح قادراً علي حل أي عددٍ نرغب في حله من المعادلات مع التحسب لكل الظروف الشاذة و الغريبة، و يجب علينا الآن أن نقوم بتحويل هذا الخوارزم إلي لغةٍ يفهمها الحاسوب ليستطيع تنفيذه.

حسنٌ: سيكون شكل البرنامج بعد كتابته بإبداع كما يلي:

\ عَرِّف متغيراً نصياً لاستخدامه لمعرفة هل يريد المستخدم حل معادلةٍ جديدة عند الانتهاء من حل الحالية، أم لا /

نص حل.معادلة.أخري = "نعم"

```
\ جملةٌ تكراريةٌ لحل أكثر من معادلةٍ من المعادلات التربيعية حسب العدد الذي يريده المستخدم /
بينما حل.معادلة.أخري = "نعم":

    \ قراءة المعاملات المختلفة من المستخدم /
    أكتب.نص.سطر("")
    أكتب.نص.سطر("أدخل المعامل الأول: ")
    رقم المعامل.الأول = أقرأ.رقم()
    أكتب.نص.سطر("")
    أكتب.نص.سطر("أدخل المعامل الثاني: ")
    رقم المعامل.الثاني = أقرأ.رقم()
    أكتب.نص.سطر("")
    أكتب.نص.سطر("أدخل المعامل الثالث: ")
    رقم المعامل.الثالث = أقرأ.رقم()

    \ اختبر كون المعامل الأول يساوي صفر ام لا؛ لتلافي القسمة علي صفر فيما بعد /
    لو المعامل.الأول = 0 :
        أكتب.نص.سطر("يرجي التنبه: لقد أدخلت المعامل الأول مساوياً للصفر، سيتم الآن حل المعادلة علي أنها من الدرجة الأولي.")
        أكتب.نص("قيمة الجذر هي")
        أكتب.رقم.سطر(- المعامل.الثالث ÷ المعامل.الثاني)
    غيره :
        \ احسب قيمة ما تحت الجذر التربيعي /
        رقم قيمة.ما.تحت.الجذر = المعامل.الثاني ^ 2 - 4 × المعامل.الأول × المعامل.الثالث

        \ هل القيمة التي تحت الجذر موجبةٌ أم سالبة؟ /
        منطق جذور.حقيقية = قيمة.ما.تحت.الجذر >= 0

        \ اكتب نص المعادلة علي الشاشة بالشكل المناسب /
```

أكتب.نص("المعادلة المطلوب حلها هي===>")

لو المعامل.الأول # 1 :

أكتب.رقم(المعامل.الأول)

أكتب.نص("س 2")

لو المعامل.الثاني > 0 :

أكتب.نص("+")

لو المعامل.الثاني # 0 :

لو المعامل.الثاني # 1 :

أكتب.رقم(المعامل.الثاني)

أكتب.نص("س")

لو المعامل.الثالث > 0 :

أكتب.نص("+")

لو المعامل.الثالث # 0 :

أكتب.رقم(المعامل.الثالث)

أكتب.نص. سطر("=0")

\ اكتب قيمة الجذرين علي الشاشة /

لو جذور.حقيقية:

أكتب.نص. سطر("الجذران حقيقيان")

رقم الجذر.الأول = (- المعامل.الثاني - قيمة.ما.تحت.الجذر ^ 0.5) ÷ (2 × المعامل.الأول)

رقم الجذر.الثاني = (- المعامل.الثاني + قيمة.ما.تحت.الجذر ^ 0.5) ÷ (2 × المعامل.الأول)

أكتب.نص("قيمة الجذر الحقيقي الأول هي")

أكتب.رقم. سطر(الجذر.الأول)

أكتب.نص("قيمة الجذر الحقيقي الثاني هي")

أكتب.رقم. سطر(الجذر.الثاني)

غيره:

أكتب.نص. سطر("الجذران تخيليان")

```
أكتب.نص("قيمة الجذر التخيلي الأول هي" + إلي.نص(- المعامل.الثاني ÷ (2× المعامل.الأول)))
أكتب.نص("+")
أكتب.نص. سطر(إلي.نص(((- قيمة.ما.تحت.الجذر) ^ 0.5) ÷ (2 × المعامل.الأول)) + "ت")
أكتب.نص("قيمة الجذر التخيلي الثاني هي" + إلي.نص(- المعامل.الثاني ÷ (2× المعامل.الأول)))
أكتب.نص("-")
أكتب.نص. سطر(إلي.نص(((- قيمة.ما.تحت.الجذر) ^ 0.5) ÷ (2 × المعامل.الأول)) + "ت")

\ اسأل المستخدم هل يريد حل معادلةٍ تربيعيةٍ أخري أم لا /
أكتب.نص. سطر("هل تريد حل معادلةٍ تربيعيةٍ أخري؟ أكتب (نعم) للقبول أو أي شيءٍ آخرٍ للرفض.")
حل.معادلة.أخري = أقرأ.نص()
```

و الآن تعال نشرح بعض الأمور التي تحتاج للشرح في البرنامج السابق:

• تعريف المتغيرات سهلٌ جداً و بسيط: يكفي أن تكتب نوع المتغير الذي ترغب فيه، ثم تكتب اسمه، ثم تكتب (= قيمة) و هذا لو أردت أن تعطيه قيمة في نفس سطر تعريفه كنوعٍ من الاختصار.

مثال 1 :

```
نص حل.معادلة.أخري = "نعم"
```

قمنا في هذا الأمر بتعريف متغيرٍ نصي (أي يمكن تخزين الحروف و الكلمات و الجُمَل داخله) و أسميناه حل.معادلة.أخري (لاحظ أن النقطة تربط بين أجزاء الإسم المنفصلة؛ لأن الأسماء يجب أن تكون من كلمةٍ واحدة). ثم قمنا بوضع النص "نعم" داخله.

مثال 2:

```
رقم المعامل.الأول = أقرأ.رقم()
```

و هنا قمنا بتعريف متغيرٍ رقمي (أي يمكن تخزين القيم الرقمية: الموجبة أو السالبة، و الصحيحة أو الكسرية داخله)، و أعطيناه الإسم "المعامل.الأول"، و بعدها طلبنا من المستخدم أن يكتب قيمةً رقميةً علي الشاشة و قرأناها عن طريق الإجراء القياسي **أقرأ.رقم**() و الذي يقرأ الرقم الذي يكتبه المستخدم ثم يعيده إلينا. و هكذا نكون قد خزّنّا قيمة المُعامِل أ في المتغير المعامل.الأول

مثال 3:

منطق جذور.حقيقية = قيمة.ما.تحت.الجذر >= 0

يعني أننا عرّفنا متغيراً من النوع المنطقي (الذي يحمل إما القيمة صحيح أو القيمة خطأ)، و أسميناه جذور.حقيقة لكي نعلم أن قيمته تدل علي كون جذور المعادلة ستكون حقيقيةً أم ستكون تخيلية، و بالطبع كان الإعتماد علي قيمة المتغير **قيمة.ما.تحت.الجذر** التي تم حسابها في تحديد هذا الأمر: بحيث لو كانت أكبر من أو تساوي الصفر (>= 0) فإن القيمة التي ستُوضع في المتغير جذور.حقيقية ستكون صحيح، و لو كانت القيمة أقل من الصفر (أي سالبة) فإن جذور.حقيقية ستكون داخله القيمة خطأ.

- سنلاحظ أن هناك كلاماً كثيراً يكون مكتوباً بين العلامتين \ / و له اللون الأخضر، و هو (كما هو واضحٌ في البرنامج) يشرح الأوامر التي تليه مباشرةً. هذا النوع يُسمي بالتعليقات؛ فهو ليس أوامراً سيتم تنفيذها و لكنه مجرد شرحٍ يكتبه المبرمج حتي يفهم الآخرون النقاط المختلفة من البرنامج، و كذلك حتي يُسهّل علي نفسه الرجوع إليه إن حدث أن تركه لفترةٍ من الفترات.

مع ملاحظة أن التعليق في إبداع يمكن أن يكون سطراً واحداً أو قد يمتد لأكثر من ذلك السطر الواحد.

أمثلة:

\ هذا تعليقٌ علي سطرٍ واحد /

\ هذا تعليقٌ علي

أكثر من

سطرٍ واحد /

•لجعل البرنامج يتكرر عدد ما نحب من المرات استخدمنا جملة **بينما** التكرارية، و التي تأتي علي الشكل التالي:

بينما قيمة.منطقية :

الأوامر.التي.نرغب.في.تنفيذها

و يمكن أن يكون للقيمة المنطقية أكثر من شكل، و هذا يُعطِي جملة بينما قوةً كبيرةً و شمولية واضحة. و في البرنامج السابق رأينا مثالاً بسيطاً علي هذه الجملة:

بينما حل.معادلة.أخري = "نعم":

و نلاحظ بالطبع أن علامة = لها معني التساؤل عن التساوي في الجُمَل المنطقية، و لم نضطر لاستخدام العلامة المُركَّبة == كما في لغات عائلة الـ C (أي لغاتٍ مثل: C، java، C++، C#)، و هذا تطبيقٌ لقاعدة استخدام الرموز و العلامات المألوفة في نفس استخداماتها في الحياة العادية. و بالإمكان ملاحظة أن أي شخصٍ عاديٍ لو قرأ سطر الكود السابق سيفهم معناه مباشرةً. في حين أنه لو قرأ سطر كودٍ كالتالي:

بينما حل.معادلة.أخري == "نعم":

فعلي الأغلب سيتساءل عن معني ==

و حينما نخبره أن معني الجملة هو(لو قيمة المتغير **حل.معادلة.أخري** تُساوِي القيمة "نعم")فسيكون التساؤل المنطقي: و لماذا غيِّرنا العلامة التي تُعبِّر عن كلمة (يساوي) و جعلناها == في حين أننا كنا نستخدم = لنُعبِّر عن كلمة (يساوي) حينما نضع قيمةً في متغيرٍ معين ؟**!**

هذه بالطبع نقطةٌ بسيطة، و لكن لو جمعناها مع شبيهاتها في إبداع لوجدنا أنها تجعل المنحني التعليمي أقل مما كان ليكون عليه لو لو نطبق مبدأ استخدام العلامات في نفس استخدامها في الواقع.

•الأمر:

أكتب.نص.سطر("مرحبا")

هو عبارة عن استدعاءٍ للإجراء القياسي **أكتب.نص.سطر** (الموجود في الإصدارات الحالية من أُبْدِع)، و يقوم هذا

الإجراء بكتابة النص الذي يتم تمريره إليه بين القوسين، ثم بعدها ينزل إلي سطرٍ جديد.

• الجملة الشرطية في إبداع تكون علي الشكل:

لو قيمة.منطقية :

الأوامر.التي.سيتم.تنفيذها

مثل:

لو المعامل.الأول = 0 :

أكتب.نص.سطر("يرجي التنبه: لقد أدخلت المعامل الأول مساوياً للصفر.")

• الأمر:

أكتب.نص("قيمة الجذر هي")

هو عبارة عن استدعاءٍ للإجراء القياسي **أكتب.نص** (الموجود في الإصدارات الحالية من أُبْدِع)، و يقوم هذا الإجراء بكتابة النص الذي يتم تمريره إليه بين القوسين، و يظل بعدها علي نفس السطر.

• الأوامر (**أكتب.رقم**) و (**أكتب.رقم.سطر**) يشبهان الأمرين (**أكتب.نص**) و (**أكتب.نص.سطر**)، لكن مع مراعاة أنهما يتعاملان مع الأرقام لا مع النصوص.

• الأمر:

لو المعامل.الثالث # 0 :

معناه (لو المعامل.الثالث لا يساوي صفر)، أي أنه في إبداع تُعبِّر علامة # عن كلمة "لا يُساوِي"، و قد استُخدِمَت لهذا الغرض لأنها تشبه و تعطِي الإيحاء الذي تعطيه علامة ≠ و التي تُستخدَم في الرياضيات للتعبير عن عدم التساوي.

يُمكِنكم أن تروا هذا في صفحة الويكيبيديا الخاصة بالرموز الرياضية:

http://en.wikipedia.org/wiki/List_of_mathematical_symbols

• في الأمر:

رقم الجذر.الأول = (- المعامل.الثاني – قيمة.ما.تحت.الجذر ^ 0.5) ÷ (2 × المعامل.الأول)

نلاحظ أنه تم استخدام العلامة ^ للتعبير عن كلمة (أُسْ)، و علامة × للتعبير عن عملية الضرب، و علامة ÷ للتعبير عن

عملية القسمة.

و بالطبع فإن ا ستخدام علامتي × و ÷ في وظيفتيهما الحاليتين بـدلاً مـن ا سـتخدام * و / لنفس الوظيفة كمـا فـي لغـات البرمجة الإنقليزية يندرج تحت قاعدة ا ستخدام المألوف من العلامات في نفس ا ستخداماتها في الحياة العا ديـة. و يجعـل تعليم اللغة للمبتدئين (و خا صة الأطفال)أ سهل.

•الأمر:

إلي.نص(100)

يُستخدم لتحويل القيمة الرقميـة إلـي قيمـة نصـية: فالمثـال السـابق يُحـوّل الرقـم 100 إلـي النـص"100" . و هـذا لـه ا ستخداماته الكثيرة في البرمجة مثل الحاجة لوضع رقم دا خل نص، و بالتـالي سـنُحوِّل الرقـم إلـي نـص ثـم نقـوم بدمـج النصين مع بعضهما البعض لتكوين نصٍ أ شمل.

•في الأمر:

أكتب.نص("**قيمة الجذر التخيلي الأول هي**" + **إلي.نص(**- **المعامل.الثاني** ÷ (2× **المعامل.الأول**)))

قمنا بدمج النصوص با ستخدام العلامة + بحيث أ صبح معني الجملة التالية:

"**قيمة الجذر التخيلي الأول هي**" + **إلي.نص(**- **المعامل.الثاني** ÷ (2× **المعامل.الأول**))

هو: تكوين نص من جملة "قيمة الجذر التخيلي الأول هي" مضافاً إليها قيمة الرقم الناتج من العملية الحسابية بعد تحويه إلي نص.

•الأمر:

حل.معا دلة.أخري = **أقرأ.نص**()

هوا ستدعاءٌ للإجراء القيا سي **أقرأ.نص** (يُوجَد في الإ صدارات الحالية من أُبْدِع)، و الـذي يقـوم بقـراءة النـص الـذي يكتبـه المُستخدِم علي الشا شة ثم إعا دته للمُبرمِج.

•لاحظوا أن الكُتَل الأمرية code blocks تُحدِّ د عن طريق الإزاحة: يعني أن الأوامر التي لها إزاحات متساوية تُعتبَر ضمن كتلة أوامرٍ وا حدة. و الأوامر التي لها تفرعات مثل جملة لو و جملة بينما يتم إزاحة أ سطر تلـك التفرُّعـات إزاحـة زائدةً عن إزاحة السطر الذي تُوجَد به كلمتي لو و بينما. و هذا مشابهٌ لما يحد ث في لغة الـ python بالضـبط (لمـن

يعلم قواعد تلك الأخيرة بطبيعة الحال).

في النهاية و بعد كل ما قلناه و شرحناه: أريدك أن تُقارِن البرنامج المكتوب بإبداع بالخوارزم المكتوب بلغةٍ عربيةٍ عادية، و قل لي:

- هل لاحظتَ أن تحويل الخوارزم إلي برنامجٍ كان سهلاً و سلساً ؟
- هل لاحظتَ أن قراءة غير الملمين بقواعد إبداع للبرنامج ستجعلهم يفهمون جانباً كبيراً منه لأنه قريبٌ جداً من اللغة المُعتادة، و أن أغلب العلامات التي استُخدِمَت في البرنامج كانت لها وظائف تشبه وظائفها في الواقع و بالتالي كان معناها مفهوماً منذ اللحظة الأولي ؟
- هل لاحظتَ أن الصعوبة كانت في الجزء الخاص بطباعة الرقم التخيلي علي الشاشة فقط؛ و ذلك بسبب خصوصية شكله؟

يمكن الحصول علي ملف البرنامج السابق من الرابط التالي علي الـ sourceforge:

http://sourceforge.net/projects/obde3/files/examples/quad.___.tar.gz/download

أو من علي الـ 4shared:

http://www.4shared.com/file/d5HnkIjW/quad.html

مع ملاحظة أنه يجب أن يكون للملف الامتداد ".رئيس" حتي تستطيع الإصدارات الجديدة من مُفسِّر "أُبْدِع" أن تترجمه و تنفذه.

المدونة الرسمية للغة البرمجة العربية "إبداع":

ebda3lang.blogspot.com

مدونتي الشخصية:

afkar-abo-eyas.blogspot.com

مدونتي العلمية:

abo-eyas.blogspot.com